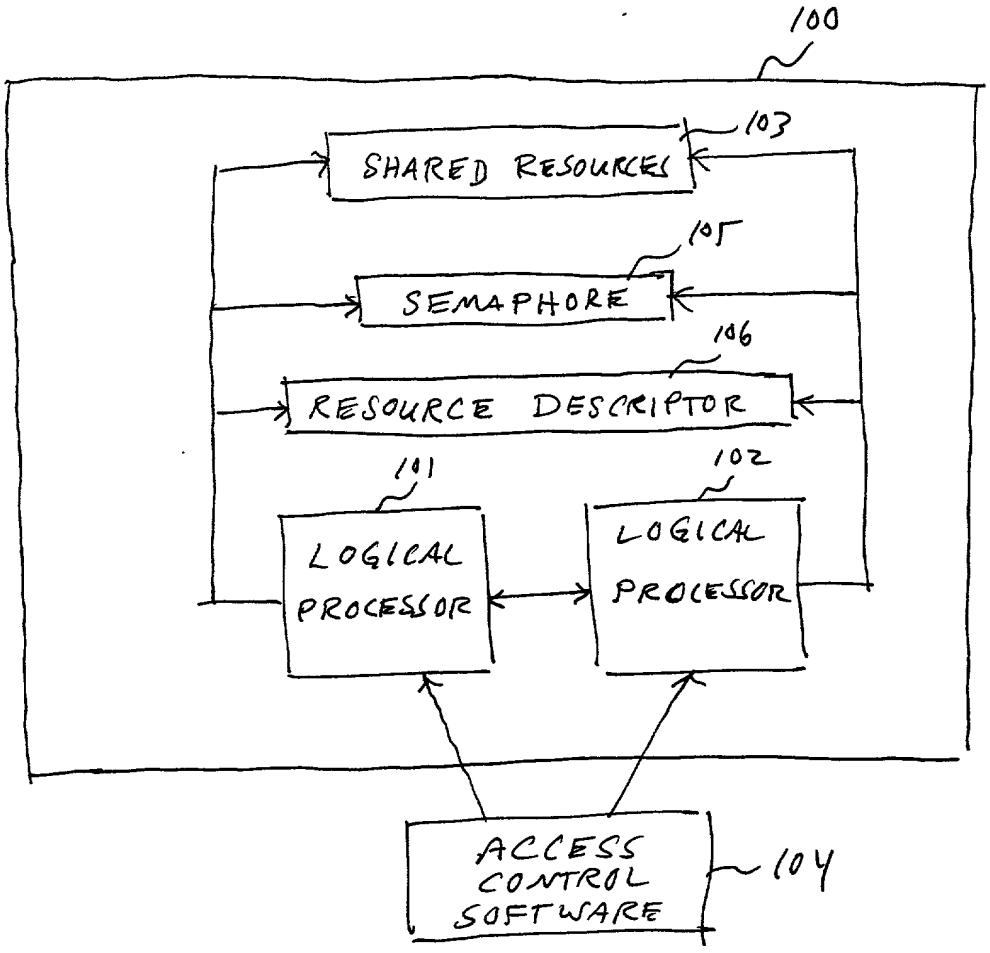
100
103
SHARED RESOURCES
105
SEMAPHORE
106
RESOURCE DESCRIPTOR
101
LOGICAL PROCESSOR
102
LOGICAL PROCESSOR
ACCESS CONTROL SOFTWARE
104

FIG. 1

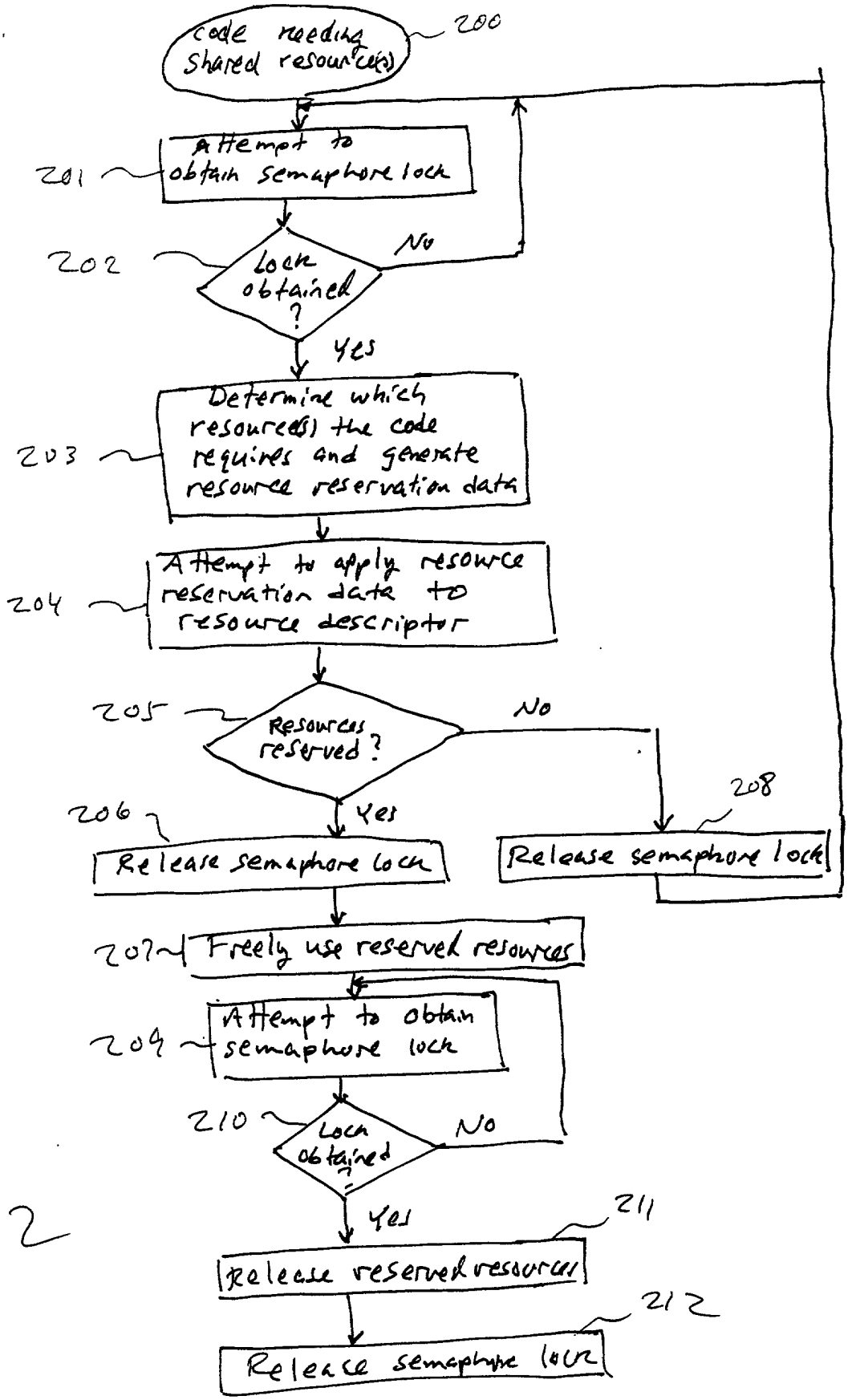
code needing shared resource(s)
200
Attempt to obtain semaphore lock
201
Lock obtained ?
202
No
Yes
Determine which resource(s) the code requires and generate resource reservation data
203
Attempt to apply resource reservation data to resource descriptor
204
Resources reserved ?
205
No
Yes
208
206
Release semaphore lock
Release semaphore lock
Freely use reserved resources
207
Attempt to obtain semaphore lock
209
Lock obtained ?
210
No
Yes
211
Release reserved resources
212
Release semaphore lock

FIG. 2

105

BIT POSITION	N 1	0
DESRCIPTION	LPID	Lock

106

Resource #	Resource 2 (Y=2)			Resource 1 (Y=1)			Resource 0	
Bit Location	Y(N+1)+N	Y(N+1)+1	Y(N+1)	Y(N+1)+N	Y(N+1)+1	Y(N+1)	N 1	0
Description	LPID		Lock	LPID		Lock	LPID	Lock

FIG. 3

106.1

Resource #	Resource 2 (y=2)		Resource 1 (y=1)		Resource 0	
Bit Location	5	4	3	2	1	0
Description	LPID	Lock	LPID	Lock	LPID	Lock

106.2

Resource #	Resource 2 (y=2)		Resource 1 (y=1)		Resource 0	
Bit Location	8 7	6	5 4	3	2 1	0
Description	LPID	Lock	LPID	Lock	LPID	Lock

106.3

Resource #	Resource 2 (y=2)		Resource 1 (y=1)		Resource 0	
Bit Location	11 9	8	7 5	4	3 1	0
Description	LPID	Lock	LPID	Lock	LPID	Lock

FIG. 4

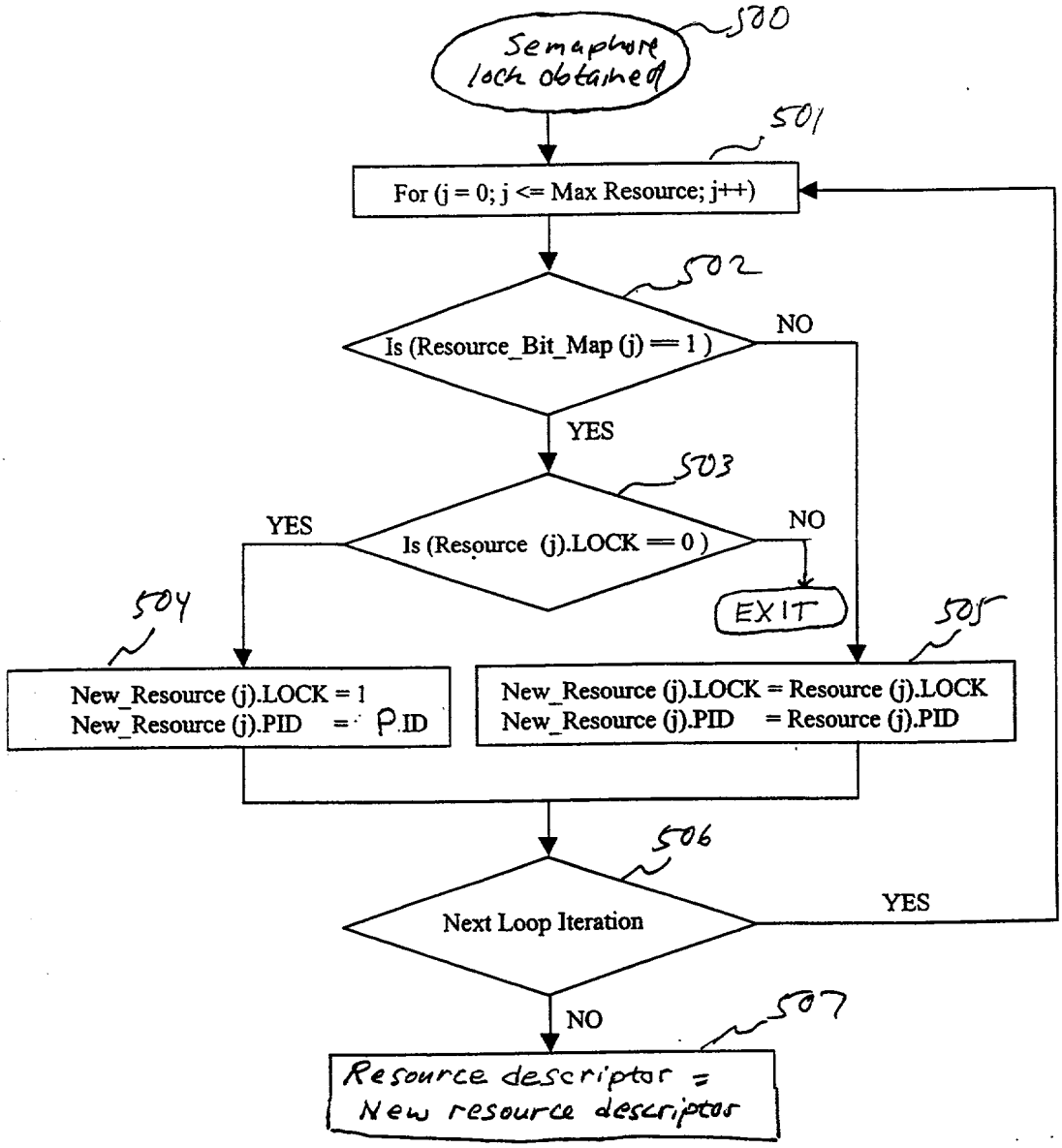
Semaphore lock obtained
500
501
For (j = 0; j <= Max Resource; j++)
502
Is (Resource_Bit_Map (j) == 1)
NO
YES
503
Is (Resource (j).LOCK == 0)
YES
NO
EXIT
504
New_Resource (j).LOCK = 1
New_Resource (j).PID = P.ID
505
New_Resource (j).LOCK = Resource (j).LOCK
New_Resource (j).PID = Resource (j).PID
506
Next Loop Iteration
YES
NO
507
Resource descriptor = New resource descriptor

FIG. 5

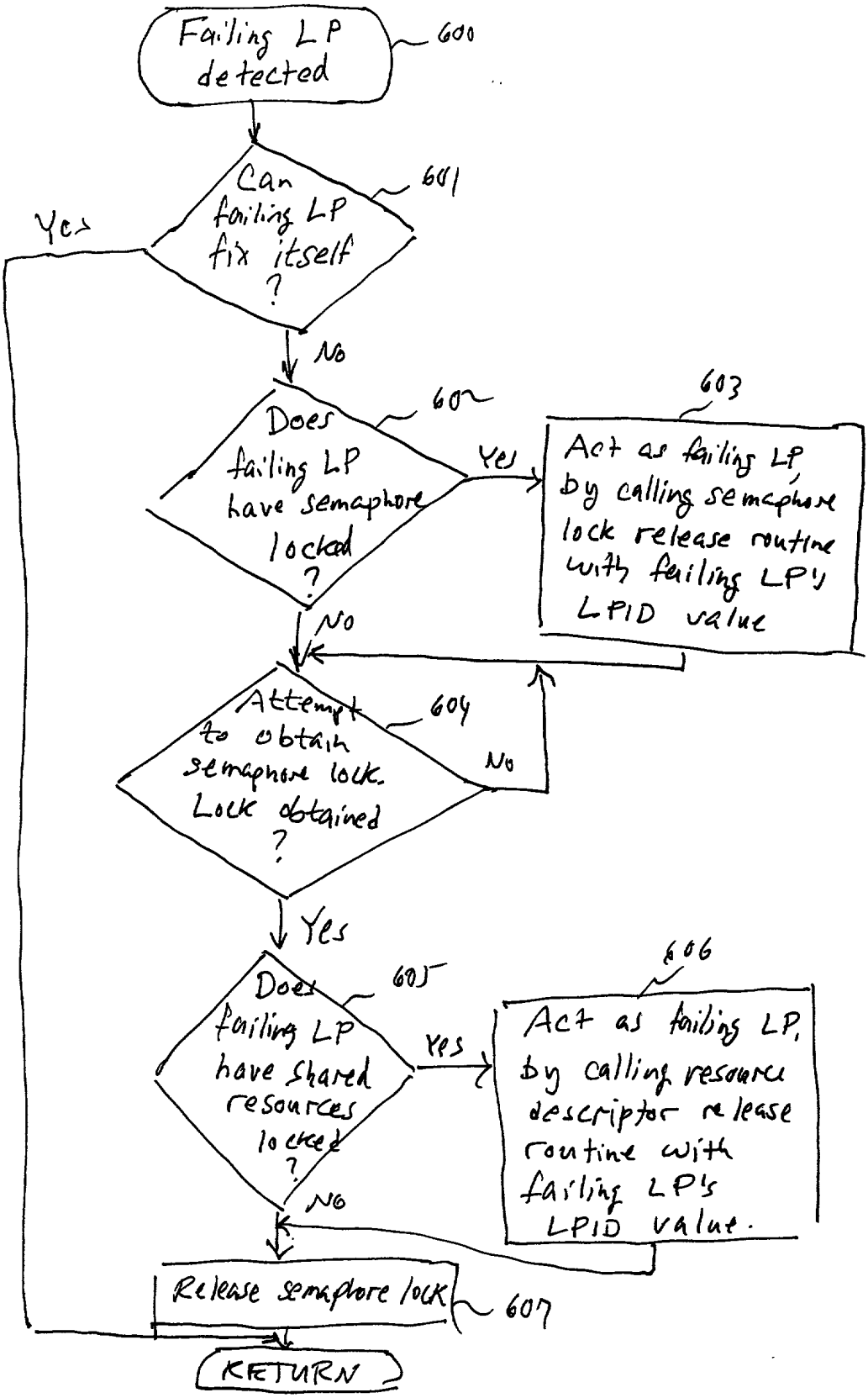
Failing LP detected
600
Can failing LP fix itself ?
601
Yes
No
Does failing LP have semaphore locked ?
602
Yes
603
Act as failing LP by calling semaphore lock release routine with failing LP's LPID value
No
Attempt to obtain semaphore lock. Lock obtained ?
604
No
Yes
Does failing LP have shared resources locked ?
605
Yes
606
Act as failing LP, by calling resource descriptor release routine with failing LP's LPID value.
No
Release semaphore lock
607
RETURN

FIG. 6

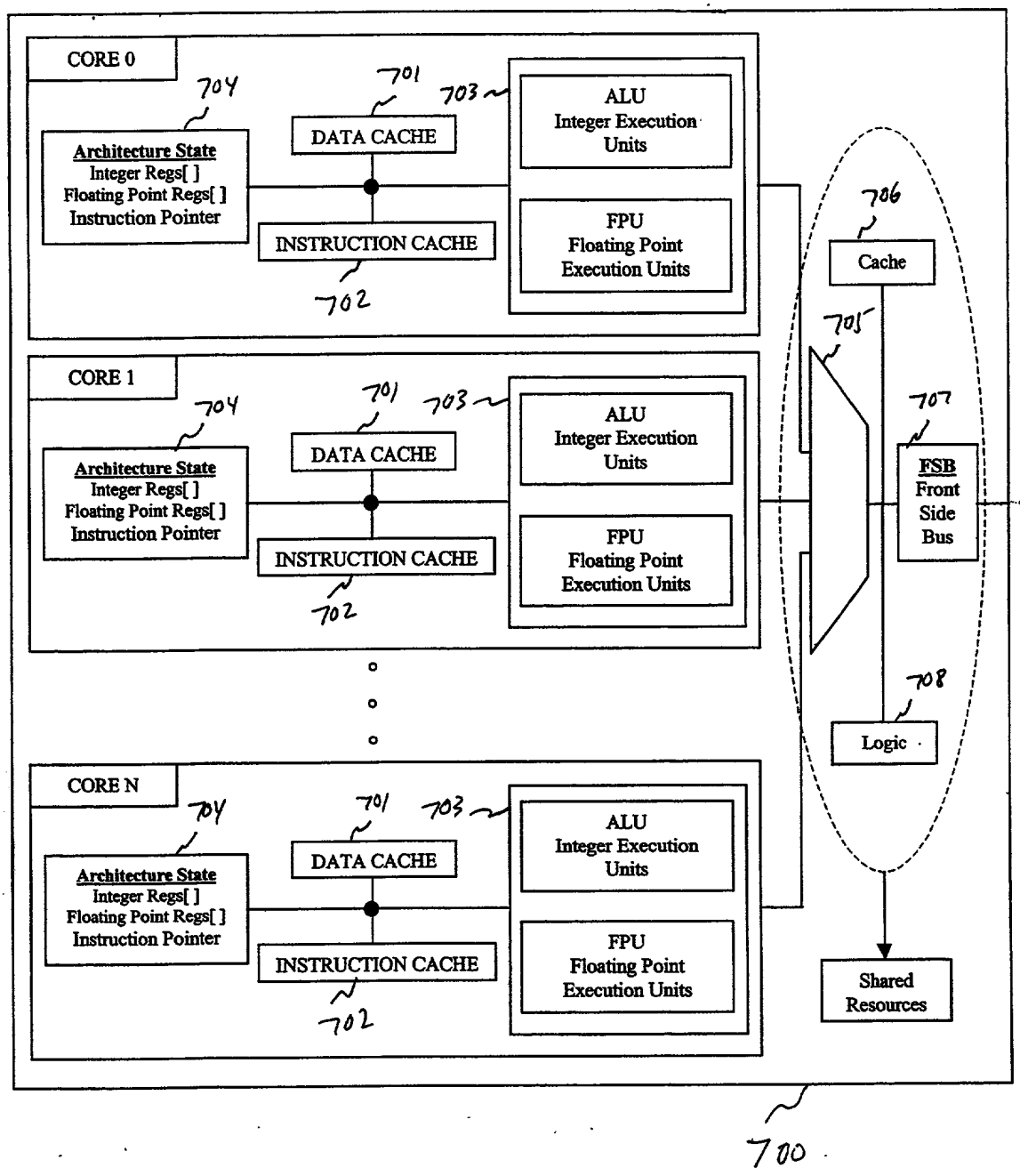
CORE 0
704
701
703
DATA CACHE
ALU
Integer Execution Units
Architecture State
Integer Regs[ ]
Floating Point Regs[ ]
Instruction Pointer
INSTRUCTION CACHE
FPU
Floating Point Execution Units
702
CORE 1
704
701
703
DATA CACHE
ALU
Integer Execution Units
Architecture State
Integer Regs[ ]
Floating Point Regs[ ]
Instruction Pointer
INSTRUCTION CACHE
FPU
Floating Point Execution Units
702
CORE N
704
701
703
DATA CACHE
ALU
Integer Execution Units
Architecture State
Integer Regs[ ]
Floating Point Regs[ ]
Instruction Pointer
INSTRUCTION CACHE
FPU
Floating Point Execution Units
702
706
Cache
705
707
FSB
Front Side Bus
708
Logic
Shared Resources
700

FIG. 7

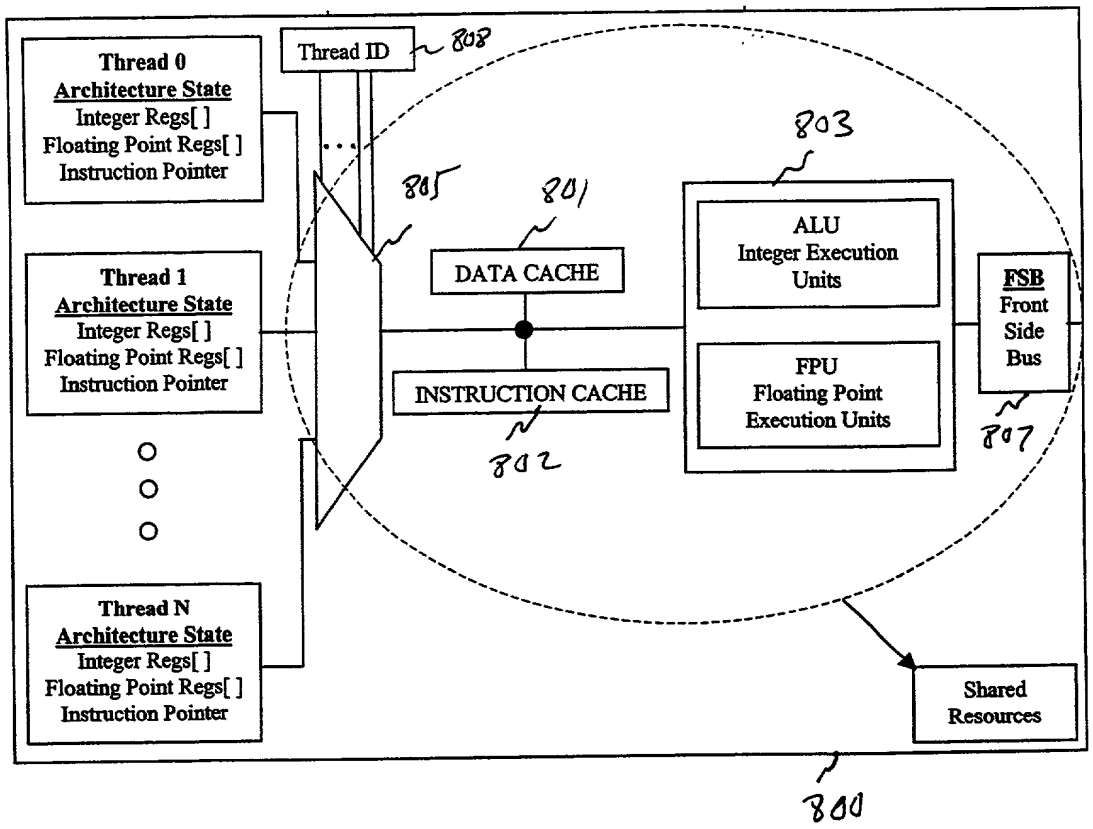
Thread 0
Architecture State
Integer Regs[ ]
Floating Point Regs[ ]
Instruction Pointer
Thread 1
Architecture State
Integer Regs[ ]
Floating Point Regs[ ]
Instruction Pointer
Thread N
Architecture State
Integer Regs[ ]
Floating Point Regs[ ]
Instruction Pointer
Thread ID
808
805
801
DATA CACHE
INSTRUCTION CACHE
802
803
ALU
Integer Execution Units
FPU
Floating Point Execution Units
FSB
Front Side Bus
807
Shared Resources
800

FIG. 8